

**REMARKS**

This Amendment is filed on March 25, 2005, within two months of the mailing date of the Final Office Action dated January 27, 2005. The Applicants thank the Examiner for her careful review of the present application.

Claims 1, 3-4, 6-8, and 10-20 are pending after entry of the present Amendment.

**Doubt Patents Rejection**

Applicants acknowledge Examiner's provisional rejection of claims 1, 3-4, 6-8, and 10-18 under the judicially created doctrine of obviousness-type doubt patenting as being unpatentable over claims 1-13 of co-pending application, Application No. 09/833,845. Applicants also acknowledge Examiner's provisional rejection of claims 19-20 under the judicially created doctrine of obviousness-type double patenting as being unpatentable over 1-13 of the co-pending application in view of Andersson, XP-002249737. A terminal disclaimer will be filed upon allowance of the pending Claims 1, 3-4, 6-8, and 10-20 in the present application.

**Rejections under 35 U.S.C. § 102(b):**

Claims 1, 3-4, 6-8, and 10-18 are rejected under 35 U.S.C. § 102(b) as being unpatentable over Ma et al. (U.S. Patent 5,920,725), hereinafter "Ma". For the reason put forth below, Applicants respectfully assert that Ma et al. fails to disclose each and every feature recited in independent claims 1, 8, and 15.

Ma discloses a run-time object-updating system that updates a distributed-object client-server application with client objects and server objects. *See col. 4, lines 36-38*. The updating system includes a central meta server on a server machine that stores object

descriptors for programming objects in a distributed client-server application. *See col. 6, lines 1-19 and Figure 3.* The meta server is a program-object cataloging program running on a server machine. *See col. 6, lines 34-35, and Figure 5.* The object class definition in the meta server serves as the blueprint for instantiating an object. *See col. 6, lines 48-51.* As shown in Figure 5, the meta server, compiler (78), and object adaptor (80) are located in server machine (90). Figure 6 shows a hierarchy diagram of a distributed application that is update by the meta server. The meta server operates on top of the server-side application. *See col. 9, lines 46-49.* Throughout the disclosure, Ma provides numerous references that the meta server, compiler (78), and object adaptor (80) operate on the server-side of a distributed two-tiered client-server application. *See Figure 5, and col. 15, lines 54-55 and 63-65.*

In contrast, as recited in claims 1, 8, and 15, embodiments of the present invention provide online upgrade of a JAVA application in the middle-tier. The middle-tier is a distinct and separate layer that is not in the server-side or the client-side of a multi-tiered network. The middle-tier is a layer that is typically located between the server and the client. The JAVA application, as recited in claims 1, 8, and 15, includes a JAVA module having an original entity bean and an original state object in the middle-tier. The original state object is in communication with the original entity bean.

Since Ma teaches updating objects in a two-tiered client-server application, Ma teaches away from updating objects in a JAVA application in the middle-tier. That is, if one of ordinary skill follows the teachings of Ma and applies the teachings to update objects in the server-client application, one would not be updating objects in the middle-tier. Updating objects in the server-client application is not the same as upgrading objects of a JAVA application in the middle-tier. As previously discussed, the middle-tier is a distinct and separate layer in a multi-tiered network that is not in the server-side or client-side of the network architecture. Thus, by following the teachings of Ma, one of ordinary skill would

not be performing the same process of object upgrade in the middle-tier and obtaining the results as claimed in the present invention.

The Examiner asserts that the server app (86) of Ma is equivalent to the original entity bean as recited in the present claims, and the rules (81) is equivalent to the original state object. *See page 5, Item 8 of the Final Office Action.* So, following the logic of the Examiner's assertion, the rules (81) are in communication with the server app (86), and rules (81) stores a state of the server app (86). However, according to Ma, rules (81) do not store any state or object class definition of the server app (86). Instead, object class definitions are stored in the meta server's non-volatile storage (62), and the compiler (78) compiles and links the class definitions to generate modified server objects and client objects (*note: the modified server and client objects of Ma are not in the middle-tier*). *See col. 8, lines 1-10.*

Moreover, workflow rules (81) are conditions that are checked after an object is updated, and the workflow rules (81) are evaluated only on the server side. *See col. 8, lines 55-57.* Thus, according to Ma, the workflow rules (81) have nothing to do with storing a state of the server app (86). Whereas, as claimed in independent claims 1, 8, and 15, the original state object store a state of the original entity bean, and the original entity bean and original state object in the JAVA module are in the middle-tier. From reading Ma, the server app (86) is not equivalent to the original entity bean, and the rules (81) is not equivalent to the original state object as recited in independent claims 1, 8, and 15.

The Examiner also asserts that Ma discloses the feature of transferring the state stored in the original state object to the updated state object as claimed in independent claims 1, 8, and 15. *See page 6, first paragraph of the Final Office Action.* Ma discloses the object adaptor (80) updating the client class definitions, and the state of an old client object is transferred to a new client object when the new client object is created from new classes. *See col. 11, lines 25-38.* In contrast, independent claims 1, 8, and 15 claim the feature of

transferring the state stored in the original state object in the middle-tier to the upgraded state object in the middle-tier. In addition, the original state object, as claimed, is in communication with the original entity bean. However, the old client object of Ma is not in communication with the rules (81). *See Figure 5. (Note: the Examiner asserts that rules (81) is equivalent to the original entity bean)*

Furthermore, the Examiner asserts that Ma discloses the claimed feature of providing state management for the original entity bean using the upgraded state object (Figure 8, 152 and 144). *See page 6, first paragraph of the Final Office Action.* Ma discloses an object adaptor (80) updating the client class definition and sending an invalidate notice to the client's object cache to invalidate an old client (144). *See col. 11, lines 25-27.* An error-handling routine in the old client object (144) determines whether an old client object (144) is invalid. The error-handling routine creates a new client object (147). The new client object (147) then re-accesses or re-creates a new server object (152). *See col. 11, lines 33-40.* As explained in Ma, the adaptor (80) updates the client class definition, and the error-handling routine in the old client object (144) indirectly creates a new server object (152) by creating a new client object (147).

In contrast, independent claims 1, 8, and 15 recite a JAVA module being executed on a server, and the JAVA module is in the middle-tier. The JAVA module includes at least one original entity bean and at least one state object. The upgraded state object provides state management for the original entity bean. The upgraded state object is an object in a middle-tier. Hence, the upgraded state object is not a client object as the old client object (144) of Ma. The original entity bean is a JAVA application, and it is also in the middle-tier. Hence, the original entity bean is not a server object as the new server object (152) of Ma.

Based on the differences discussed, independent claims 1, 8, and 15 are not anticipated by Ma under 35 U.S.C. § 102(b). Similarly, dependent claims 3-4, 6-7, 10-14,

and 16-18, drawing their respective dependencies from either independent claim 1, 8, or 15 are not anticipated by Ma for substantially the same reason as discussed, and for the additional limitations in which each dependent claim respectively recites.

**Rejections under 35 U.S.C. § 103(a):**

Claims 19-20 are rejected under 35 U.S.C. § 103(a) as being unpatentable over Ma as applied to claims 15-18, and further in view of Andersson (XP-002249737), hereinafter "Andersson". Applicants respectfully traverse the Examiner's rejection.

As discussed in the previous section, Ma discloses a run-time object-updating system that updates a distributed-object in a two-tiered client-server application with client objects and server objects. *See col. 4, lines 36-38*. As discussed in the previous section, Ma fails to disclose upgrading managed state for a JAVA based application in a middle-tier as claimed in independent claims, 1, 8, and 15. Thus, even if Ma is combined with Andersson, a proposition that the Applicants would traverse (*see response to the previous non-final Office Action*), the combination of Ma and Andersson still fails to disclose each and every feature of dependent claims 19 and 20 to render the claims obvious.


That is, even if Andersson discloses the teaching of classifying objects into state management units, a proposition that the Applicants would traverse, the combination of Ma and Andersson still fails to disclose all the features of dependent claims 19 and 20 including all the limitations of the intervening claims and independent claim 15 to render claims 19 and 20 obvious.

Accordingly, after entry of the present Amendment, the application is now in a condition for allowance. A Notice of Allowance is therefore respectfully requested.

If the Examiner has any questions concerning the present Amendment, the Examiner is kindly requested to contact the undersigned at (408) 774-6911. If any other fees are due in

connection with filing this Amendment, the Commissioner is also authorized to charge  
Deposit Account No. 50-0805 (Order No. SUNMP007). A duplicate copy of the transmittal  
is enclosed for this purpose.

Respectfully submitted,  
MARTINE PENILLA & GENCARELLA, LLP

  
William K. Yee, Esq.  
Reg. No. 54,943

710 Lakeway Drive, Suite 200  
Sunnyvale, CA 94085  
Telephone: (408) 749-6900  
Facsimile: (408) 749-6901  
Customer No. 25920